AmyIRC

Michael van Elst

COLLABORATORS

	TITLE : AmyIRC	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Michael van Elst	August 5, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

AmyIRC

1.1 AmyIRC

```
  AmyIRC - a small Amiga IRC client for
AS225
 and
AmiTCP

 .
 - What is IRC

 .
 - How to start AmyIRC

 .
 - The Parameters

 .
 - How to enter Commands

 .
 - AmyIRC Commands


 .
 - Known Bugs

 .
 - Pointers to the Software used


 .
 - Credits
```

1.2 C= TCP/IP Networking Package AS225 Release 2

```
AS225 is a TCP/IP protocol stack for the Commodore Amiga.
```

The first release used built-in drivers for only a few network interfaces like the A2065 Ethernet card. Also Commodore did not publish the API for this stack so you couldn't write your own client software.

A second release of AS225 has been in beta-testing for quite some time.

At this time you still cannot buy it, but registered Amiga developers may use it.

AmyIRC
 is written for AS225R2, but also works with the Public Domain
AmiTCP
 protocol stack.

1.3 Amiga TCP protocol stack

AmiTCP is a free TCP/IP protocol stack.

Since Commodore has not, up to now, supplied a versatile networking package for the Amiga some people in Finland started to create their own software based on the Berkeley Net/2 code. This software is called AmiTCP and is now considered the only solution to network Amigas with Unix computers and the world wide Internet.

There is a continous effort to add services and clients to AmiTCP. One important item developed for AmiTCP is a

socket.library
 that translates the
AS225
 API to the slightly different AmiTCP API.

This library makes it possible to use clients and services that are compiled for the AS225 package to work under AmiTCP, e.g.

AmyIRC
.

AmiTCP is published under terms of the GNU PUBLIC LICENSE. For more information on AmiTCP you can reach the authors from the e-mail address <AmiTCP-Group@hut.fi>.

1.4 socket.library emulation

The socket.library

Each software library has a set of user-callable functions and accessible data structures called the API (Application Programmers Interface). The two Amiga TCP/IP protocol stacks

AS225

and
AmiTCP
use an API very similar to
the BSD socket model but varying slightly in how they integrate this model
into the AmigaOS framework.

Since both protocol stacks are widely used, you would need a special program
version for either
AS225
or
AmiTCP
. While this problem may cease to exist
in the future (as Commodore has stopped any development on
AS225
) there is
still some demand for applications that work on both stacks. One solution is
to emulate one API with the other.

The socket.library by Henning Schmiedehausen tries to emulate the
AS225
API.
While the emulation isn't perfect it is complete enough to use
AmyIRC
together with
AmiTCP
.

1.5 The Internet Relay Chat

The Internet Relay Chat

IRC is the second largest waste of productivity invented in the computer
industry so far (the largest one is the World Wide Web which is sufficiently
easy to be used even by computer illiterates}.

IRC is also a very successful communication method that connects computers
on the world's largest computer network called the Internet. In IRC discussions
are organized into channels where people with related interests talk
about god and the world (and also about their computers).

People on IRC use
nicknames
to address each other. But in many situations
it is good when you also know the
real person
who stands behind a nickname.

Since communication through the IRC system is very fast it can react quickly
to current events. In the beginning of the Gulf War people gathered in the
#report channel and exchanged news and reports from participants much faster
than the official news agencies could deliver information.

1.6 How to start AmyIRC

```
              AmyIRC
               is an Amiga IRC client modelled after the UNIX Irc II program.
It is a shell-based program as opposed to a program with a graphical
user interface. Thus it is possible to use AmyIRC through any sufficiently
powerful emulation of the Amiga
              CONsole
              , especially through a remote
telnet session or as an external service to some BBS programs.

If you want to run AmyIRC under
              AmiTCP
               you need to install the
              socket.library
                first. This library is an emulation of the
              AS225
               API library with the
same name.
```

1.7 Parameters

```
              AmyIRC
               uses a standard AmigaDOS ReadArgs template:

IRC IRCRC <filename> NICK <irc nickname> LOGIN <irc loginname>
    REALNAME <your real name> PASS <server password> PORT <tcp/ip port number>

Most options can be preset by a AmigaShell variable or Environment variable,
i.e.

Variable       Parameter           Default
USERNAME     -
              LOGIN
                              "AmyIRCer"
REALNAME     -
              REALNAME
                           "Amiga IRC User"
IRCNICK      -
              NICK
                               "AmyIRC"
IRCSERVER    -
              SERVER
                            "irc.rz.uni-karlsruhe.de."
IRCPASS      -
              PASS
                               none
IRCPORT      -
              PORT
                               6667
IRCRC        -
```

IRCRC
"s:ircrc"

1.8 LOGIN, REALNAME and NICK parameter

An IRC client presents three identifiers to the IRC servers. All of these are arbitrary but certain sites may require that you supply true values.

LOGIN is the name of the account on your machine.

People assume that they can send e-mail to you at LOGIN@host.domain where the host and domain name is the Internet address of your Amiga generated from your Internet number. If your Amiga does not accept mail you should ask your network provider for a MX entry that reroutes mail to your mailbox on a different host.

REALNAME is your name in real life.

Some IRC operators require that you do not hide under a fictional name. However on most sites pseudonyms are accepted as long as people can get a real id on you (e.g. through your LOGIN).

NICK is your nickname or handle.

All IRC participants are addressed by a nickname that each chooses and which may be up to 9 characters in length. Nicknames must be unique.

1.9 About Nicknames

How to choose nicknames?

Good examples for nicknames are DrDelete, ElecWnch or Deli (but read on..), bad nicknames are DrWho, Jim or Idiot.

One constraint is that you cannot use a nickname that is currently used by someone else, nicknames must be unique. It also causes lots of trouble when you choose a nickname that is used frequently by other people since then they are forced to use a different nickname.

Now why are the first three nicknames good and why you must not use them?

These nicks aren't any popular or common words or names. The risk of nickname conflicts is minimal. They are also unique enough that people will remember you when you join IRC the next time. These nicks are also abstract enough so that people aren't led to make fun of your nickname

(and yourself). Now, these nicknames are still bad for you, simply because
in this example I used nicknames that are already used by other people.

It should now be obvious why the last three nicknames are not good choices
either. But I am pretty sure that you will find your very own NICK.

1.10 SERVER parameter

The
IRC
system builds its own network of clients and servers.
AmiTCP
is an IRC client which connects to a server machine that routes ↩
all
messages and operations to their final destinations.

The default server for
AmyIRC
is irc.rz.uni-karlsruhe.de. which is a
site in the middle of Germany. While this site is good for many users
it is much better and faster to use the nearest server to your site.
If your network provider runs an IRC server then this is the best
machine for you to connect.

AmyIRC accepts either an Internet number or an Internet address for the
SERVER parameter. And most people will probably prefer to use a Shell-
or Environment-Variable to preset the server for all IRC sessions.

1.11 PASS and PORT parameters

IRC
uses the TCP/IP protocol to create links between clients and ↩
server
sites. TCP/IP uses a port number to identify such network connections. While
the client uses an arbitrary private port the server must use a specific
port that the client can refer to. Most IRC servers use the port number
6667 which is also the default for
AmyIRC
but some installations may use
different port numbers. In that case use the PORT parameter to supply a
numeric port number.

A very few server sites restrict access to smaller groups. On those sites
access is granted by supplying a password after connecting to the server.
I don't think that you will ever need that feature, but if you have to you
can give a password with the PASS parameter.

1.12 IRCRC parameter

Startup

AmyIRC can read a startup file that contains commands that are executed directly after a connection to the IRC server is established. The file contents is passed literally to the command interpreter line by line. This lets you customize your IRC session by adding your own command aliases and function key definitions. You can also use the startup file to automatically log into your prefered IRC channel(s).

Since AmyIRC is just a subset of the more powerful UNIX Irc II program there are no smart scripting capabilites and you cannot create complex commands. We plan to add AREXX support to AmyIRC in the future AREXX will also be used to create direct client connections (DCC) and client specific commands (CTCP).

1.13 Amiga CONsole

The Amiga handles most text input and output through a console window that uses standard ANSI control sequences. Since AmyIRC is designed to run in an Amiga console it can be used with any I/O stream that simulates an ANSI terminal and resembles the Amiga CON-Handler.

In particular the stream handler must allow for WAITFORCHAR packets since AmyIRC uses this packet for non-blocking reads on the input stream so that you can safely terminate the program at any point.

Normally all Amiga console handlers do support WAITFORCHAR but there might be exceptions. One example is the internal console handler of Willi Langeveld's telnetd for AS225.

AmyIRC uses several peculiar control sequences that an ANSI compliant device will either understand or ignore. These are the

- WINDOW STATUS REQUEST
- WINDOW BOUNDS REPORT

which are used to determine the console size. If the stream never sends
a WINDOW BOUNDS REPORT then AmyIRC assumes a standard terminal size
of 24 by 80 characters. And

· SET RAW EVENTS
· RESET RAW EVENTS
· INPUT EVENT REPORT

are used to learn about changes of the window size and whether the user
clicked on the window's close gadget.

1.14 Input line

AmyIRC
is a shell based program which means that it uses the Amiga

CONsole
for input and output. If you want to run AmyIRC in its own
window you can redirect its both input and output streams to a new
console. You need to redirect both streams to the same window with
the <> option of the AmigaShell like:

IRC <>"CON:10/10/600/100/Internet Relay Chat/AUTO/CLOSE"

After AmyIRC has established a connection to the IRC server the console
is split into two parts, a vertically scrolling output region and a
horizontally scrolling input line. Both parts are separated by a status
line which is rendered in inverse.

AmyIRC turns the console into RAW mode to catch every keypress while
concurrently maintaining the output region. However, most of the line
editing has been adopted by AmyIRC. You can use the cursor keys to move
the insert point, insert or append new characters, delete the character
to the left with backspace and the character under the cursor with del.

The cursor up and down keys will access the
input history
and it also
understands some
function keys
.

1.15 Input history

Input history

Moving the cursor up and down will retrieve lines from the input
history buffer which you can search for recent entries with shift

cursor up just like with the shell. In addition the current active edit buffer will be kept unless you start editing a line from the history. Thus it is possible to go back into the history, capture words to the clipboard and then go back to the active input line to insert the material from the clipboard.

1.16 function keys

The line editor has a special handling for certain function keys ↩
.
Function keys (F1 to F10, with or without shift and the HELP key) are handled differently than normal keys in that they will generate complete commands that are send to the command interpreter. They will not change the contents of the input buffer so that you can generate meta commands, e.g. to move input focus between several channels or to run external commands on a single keypress when the

AREXX
support has been added in the future.

Function keys send strings to the command interpreter. While these strings are not

AmyIRC
commands themselves they can be
aliased
to valid commands.

· The function keys F1 through F10
will generate the commands /FUNCTION1 through /FUNCTION10.

· The shifted function keys F1 through F10
will generate the commands /SFUNCTION1 through /SFUNCTION10.

· The HELP key will generate the command /HELP.

1.17 AREXX

AREXX is a scripting language that can be used to integrate several applications into a larger set and to create complex automated commands.

AmyIRC
will support AREXX in the future.

1.18 IRC commands

AmyIRC commands work similar to Irc II commands. A command starts with a lead-in character which is currently either a '/' or a '.'. Input lines that do not start with a lead-in character are treated as literal text entries that are sent to the current channel.

ADMIN

ALIAS

AWAY

BAN

CHOP

CTCP

DCC

INVITE

JOIN

KICK

LIST

LUSERS

LEAVE

MODE

ME

MSG

NAMES

NICK

NOTICE

PART

QUERY

QUIT

QUOTE

SHOW

TOPIC

UNBAN

UNCHOP

WHO

WHOIS

1.19 Admin command

/ADMIN server

Ask a server who is responsible for it.

1.20 Alias command

/ALIAS name value

Create a shorthand for another command. The new command '/name' gets replaced by 'value', any parameters of the command are appended.

It is not possible to choose an alias name that is identical to a built in command. It is possible to create an alias for an existing alias.

1.21 Away command

/AWAY {notice}

Mark yourself as away. People that query you with a WHOIS or that try to send a MSG will receive your notice.

If you use the AWAY command without arguments then the notice is removed.

See also:

MSG

WHOIS

1.22 Ban command

```
                /BAN  {nick!user@host}

Add the specified user to the current channels ban list. The ban entry
consists of a nickname nick and an user specification user@host.

Both parts can contain asterisks and question marks as wildcard characters. I.e ↩
    .:

nick!*@*  bans a specific nick whatever account is used.

*!user@host bans a specific account whatever nick is used.

*!*@*.edu bans all US universities.

Using BAN without parameters displays the channels ban list.


This command requires channel operator status.

See also:

              UNBAN

              MODE
```

1.23 Chop command

```
                /CHOP nick

Grant the user with a given nick channel operator status.


To grant channel operator status you need to be channel operator yourself.

See also:

              UNCHOP

              MODE
```

1.24 CTCP command

```
/CTCP nick command {arguments}

Send a command to the user with the given nick through the
Client-To-Client-Protocol.

What commands are possible depend on the client used. Most
clients accept
```

```
VERSION   send back a text describing what client is used.
USERINFO  some information about nick.
FINGER    the output of the finger command on nick's site.
PING    the network response time to nick.

AmyIRC currently just responds to the VERSION command.
```

1.25 DCC command

```
                /DCC  ???

DCC establishes a direct client-to-client connection which can be
used for private talks or data transfers that do not pass the
              IRC
                servers network.


Currently AmyIRC does not support DCC.
```

1.26 INVITE command

```
                /INVITE {#channel} nick

Tell the user nick that he may join your current channel or the channel
given as an argument. This is needed for channels set to "invite-only"
with MODE +i.

You need to be channel operator to INVITE someone to an invite-only
channel.

See also:

              MODE
```

1.27 JOIN command

```
                /JOIN #channel {key}

Enter a channel. You can enter up to 10 channels concurrently by sending
several JOIN commands.

There is the notion of a current channel which is the one you are talking
to. Public messages are send to this channel and messages coming from
other channels are prefixed by the channel name. I.e.:

<nick> Hello world             is a public message on the current channel
<#foo:nick> Hello world        is a public message on channel #foo
```

If you JOIN a channel that you are already on then this channel becomes the current channel.

Joining a channel may require you to use a key word if the channel uses MODE +k.

See also:

PART

MODE

1.28 KICK command

/KICK {#channel} nick

Remove the user nick from the current channel or the channel given as an argument.

You need to be channel operator for that channel to remove an user.

See also

BAN

1.29 LIST command

/LIST {#channel}

List the number of users and topic of the current channel or the channel passed as an argument.

The channel argument can contain wildcard characters, then any channel with a matching name will be listed.

If the list is too long then your IRC server may reject you.

See also:

NAMES

WHO

TOPIC

1.30 LUSERS command

```
                /LUSERS {server}

Query your server or the given server for some statistics about
the
              IRC
               network.
```

1.31 MODE command

```
                 /MODE nick [+|-]modechars {arguments}
/MODE #channel [+|-]modechars {arguments}

Alters all the settings in a session. You can change channel specific
options and those for the user nick. Options are enabled with a + and
disabled with a - in front of the mode character. Here is a list of all
modes. Changing the channel mode requires channel operator privilege.

nick     [+|-]i   make yourself invisible to WHO and NAMES except
      for users that are on your channel(s).
nick     [+|-]o pass  make yourself IRC operator.
nick     [+|-]s   tell the server that you want to receive administrative
      messages.
nick     [+|-]w   tell the server that you want to receive WALLOPS messages.

#channel [+|-]b nick  ban a user from a channel (» BAN).
#channel [+|-]i   make the channel invite-only
#channel [+|-]k key set the channel key, only people with a correct key may
      enter the channel (» JOIN).
#channel [+|-]l limit limit the number of users on a channel.
#channel [+|-]m   make the channel moderated, just channel operators and
      other privileged users (» +v) may send public messages
      to a channel.
#channel [+|-]n   disallow public messages to the channel from people on
      the outside.
#channel [+|-]o nick  make user nick a channel operator.
#channel [+|-]p   make a channel invisible to LIST and NAMES. You can
      still query it with WHO if you know the exact channel name.
#channel [+|-]s   make a channel completely secret. It will not appear on
      any list or query.
#channel [+|-]t   disallow normal users to change the channel topic. Only
      channel operators may do that.
#channel [+|-]v nick  Grant user nick to speak into a moderated channel (» +m).


See also:

              WHO
```

NAMES

BAN

CHOP

INVITE

JOIN

1.32 ME command

/ME action

You can send special action messages to your current channel. These are actually public messages with some control characters that are interpreted by newer
IRC
clients.

It is generally difficult to express emotions in written messages, using the ME command helps around this problem by using non-direct speech.

An action appears on the channel like

#channel=> nick does this and that

See also:

MSG

QUERY

1.33 MSG command

/MSG #channel message
/MSG nick message

Public messages can be read by anyone on the current channel. With MSG you can send messages to an arbitrary channel unless that channel explicitely forbids messages from the outside.

When you specify a nick as the recipient then the message is just send to the specific user. You can use a comma-separated list of nicks. In that case AmyIRC will repeat the message for each user on the list.

See also:

QUERY

ME

MODE

NOTICE

1.34 NAMES command

/NAMES {#channel}

NAMES lists all users on some channel by their nickname.

You can use wildcard characters so that all users on all matching channels are listed. If the list is too long then your IRC server may reject you.

See also:

LIST

WHO

TOPIC

1.35 NICK command

/NICK nick

Change your nickname to nick. If that nickname is used by someone else you get an error message and your nickname remains the same.

1.36 NOTICE command

/NOTICE #channel message
/NOTICE nick message

About the same as the MSG command except that the recipient(s) can distinguish between the two methods.

This was introduced to avoid infinite message loops when two or more
Irc II scripts respond to each other. By convention (and public
punishment) you are obliged to use NOTICE for any automated response
generated by a script and to ignore any NOTICEs sent to your script.

AmyIRC does not yet support scripts.

See also:

MSG

1.37 PART and LEAVE commands

/PART {#channel}
/LEAVE {#channel}

Leave the current channel or the channel given as an argument. If you have
JOINed other channels then one of these channels becomes the new current
channel.

PART and LEAVE are synonyms for some historical reason.

See also:

JOIN

LIST

1.38 QUERY command

/QUERY nick
/QUERY #channel
/QUERY

Redirect all your public messages to the given nick or channel. This comes
handy when you want to temporarily talk to another user without typing
lots of MSG commands.

QUERY without an argument removes the redirection and public messages are
sent to the current channel again (if there is any).

See also:

MSG

1.39 QUIT command

```
                /QUIT {message}

Exit from the
             IRC
              session. This informs the server that you are about
to close your network connection. The optional message is sent to the
channel instead of the default message (Leaving).

AmyIRC will wait for the final server response and then exit.
```

1.40 QUOTE command

```
/QUOTE  text

The text will be send literally to the server. With QUOTE it
is possible to use server commands that are not implemented or
accessible from a client. I.e.

/QUOTE PONG some.server.name

tells the server that your session is still alive.
```

1.41 SHOW command

```
/SHOW something

Display the information gathered so far for something. Currently
the only useful entity is "state" which shows information about
your session.
```

1.42 TOPIC command

```
                /TOPIC  {#channel} {topic}

Set the topic for the current channel or the channel passed as an argument.
If you do not specify a topic then the current topic of the channel is displayed ↩
    .


See also:

             NAMES

             WHO

             LIST
```

1.43 UNBAN command

```
/UNBAN  nick!user@host
```

Remove the specific user from the current channels ban list. The ban entry consists of a nickname nick and an user specification user@host.

You need to pass the exact ban entry that is in the channels ban list. You cannot use wildcards to remove multiple entries.

This command requires channel operator status.

See also:

BAN

MODE

1.44 UNCHOP command

```
/UNCHOP nick
```

Deny channel operator status to the user with a given nick.

To deny channel operator status you need to be channel operator yourself.

See also:

CHOP

MODE

1.45 WHO command

```
/WHO  nick
/WHO  {#channel}
```

Show information about the user nick or all users on a given channel.

Both specifications can use wildcard characters to match multiple nicks or channels. Using WHO with no argument lists all users from the current channel.

See also:

WHOIS

LIST

NAMES

1.46 WHOIS command

/WHOIS nick

Query information about a specific nick. Using WHOIS is the only method to query users that made themselves invisible with the MODE command. However, you need to know their exact nick as WHOIS will not accept wildcards.

See also:

WHO

MODE

1.47 Bugs, are there any?

BUGS

AmyIRC is still in a very early state. So are there bugs? Obviously. Do we know them? Well, some.

- AmyIRC does not support the clipboard correctly under AmigaOS 2.0. In fact, it doesn't handle input from the clipboard at all but relies on the smarter AmigaOS 3.0 console.device to insert clipboard material into the input stream.

- AmyIRC does not run under AmigaOS1.3. No, this is definitely not a bug. It is a worthwhile feature since it will lead people to get rid of this old OS release that has been obsolete for quite some time.

1.48 Pointers to software

Most people that want to use AmyIRC probably need a working AmiTCP setup and the socket.library

.

Both can be found on the Aminet cluster of anonymous ftp servers in the subdirectory comm/net.

If you don't know any Aminet site near to you then ftp to ftp.wustl.edu (also known as wuarchive.wustl.edu) and get the file /pub/aminet/README. It will tell you more about this archive of freely distributable Amiga software and contains a list of all sites that mirror the Aminet archive.

1.49 Credits

AmyIRC
was created by

· Markus Illenseer ("ill")
Initial main loop, user interface, resource tracking

· Matthias Scheler ("tron")
Networking code

· Michael van Elst ("mlelstv")
Shell interface, console, networking, SASification

· S. P. Zeidler ("stargazer")
Command parser, state machines, commands code

Special thanks to Henning Schmiedehausen ("barnard") for the
socket.library
, all people from the AmiTCP-Group,
and Oleg Rovner ("oleg") and Bernhard Möllemann ("ZZA") for proof-reading this guide.